Text Technologies for Data Science
INFR11145

Web Search (2)

Instructor:
Walid Magdy

Lecture Objectives

• Learn about:
  • Basics of Web search
  • Brief History of web search
  • SEOs
  • Web Crawling (intro)
**Brief History**

  - Altavista, Excite, Infoseek, Lycos
  - Traditional IR techniques
  - Scalability is an issue

- **Paid search** ranking: Goto (morphed into Overture.com → Yahoo!)
  - Your search ranking depended on how much you paid
  - Auction for keywords
  - Called “sponsored search”
    - CPC (Cost Per Click)
    - CPM (Cost Per Thousand Impressions)

**Brief (non-technical) History**

- 1998+: Link-based ranking pioneered by Google
  - Blew away all early engines
  - Great user experience in search of a business model
  - Meanwhile Goto/Overture’s annual revenues: ~ $1 billion

- Result: Google added paid search “ads” to the side, independent of search results
  - Yahoo followed, acquiring Overture (for paid placement) and Inktomi (for search)

- 2005+: Google gains search share, dominating in Europe and very strong in North America
  - 2009: Yahoo! and Microsoft combined paid search offering
Web Search Basics

User

Web spider

Indexer

Indexes

Ad indexes

Walid Magdy, TTDS 2018/2019
User Need on Web Search

- **Informational** – want to learn about something (~40% / 65%)
  - Information Retrieval

- **Navigational** – want to go to that page (~25% / 15%)
  - United Airlines

- **Transactional** – want to do something (web-mediated) (~35% / 20%)
  - Access a service: Seattle weather
  - Downloads: Mars surface images
  - Shop: Canon S410

- **Gray areas**
  - Exploratory search “see what’s there”

Search Engine Optimization (SEO)

- The Trouble with Paid Search Ads:
  It costs money. What’s the alternative?

- **Search Engine Optimization (SEO):**
  - “Tuning” your web page to rank highly in the algorithmic search results for selected keywords
  - Alternative to paying for placement
  - Thus, intrinsically a marketing function

- Performed by companies, webmasters and consultants (“Search engine optimizers”) for their clients

- Some perfectly legitimate, some very shady
SEO: Simplest Form

- First generation engines relied heavily on \( tf/\text{idf} \)
  - The top-ranked pages for the query \textit{maui resort} were the ones containing the most \textit{maui’s} and \textit{resort’s}.
- SEOs responded with dense repetitions of chosen terms
  - e.g., \textit{maui resort maui resort maui resort}
  - Misleading meta-tags, excessive repetition
  - Often, the repetitions would be in the same color as the background of the web page
    - Repeated terms got indexed by crawlers
    - But not visible to humans on browsers

*Pure word density cannot be trusted as an IR signal*

SEO word manipulating examples

- XYZ Hotel in ABC city
  - Accommodation, hotel, room, flat, travel, sights, attractions, vacation, holiday, in ABC ABC ABC
- XYZ Umbrellas
  - Raining, rainy, wet, weather, day
- XYZ for family advices
  - Family, couples, parents, spouse, wife, husband, fights, relationship, cheating, communication, kids, children
**SEO: Cloaking**

- Serve fake content to search engine spider
- Famous technique: **Black Hat**
- Kind of a spam!

**Duplicate Detection**

- The web is full of duplicated content
- Strict duplicate detection = exact match
  - Not as common
  - can be detected with fingerprints
- But many, many cases of **near duplicates**
  - e.g., last modified date the only difference between two copies of a page
- **Near-Duplication: Approximate match**
  - Use similarity threshold to detect near-duplicates
    - e.g., Similarity > 80% => Documents are “near duplicates”
    - Not transitive though sometimes used transitively
      - $A \approx B$ & $B \approx C \rightarrow$ doesn’t have to mean $A \approx C$
Duplicate Detection: MiniHash

• Features of similarity:
  • Segments of a document (natural or artificial breakpoints)
  • **Shingles** (word n-grams)
  • *a rose is a rose is a rose* →
    
    a_rose_is_a  
    rose_is_a_rose  
    is_a_rose_is  
    a_rose_is_a

• Similarity measure between two docs (= sets of shingles)
  • Set intersection
  • Specifically (Size of Intersection / Size of Union)

---

Shingles + Set Intersection

• Computing exact set intersection of shingles between all pairs of documents is expensive/intractable

• Approximate using a cleverly chosen subset of shingles from each (a sketch)

• Estimate \( \frac{\text{size of intersection}}{\text{size of union}} \) based on a short sketch
Web Crawling

Basic Crawler Operation

• Begin with known “seed” URLs
• Fetch and parse them
  • Extract URLs they point to
  • Place the extracted URLs on a queue
• Fetch one URL from the queue
• Repeat
What Any Crawler Must Do

• Be **Polite**: Respect implicit and explicit politeness considerations
  • Only crawl allowed pages
    • respect `robots.txt`
  • Avoid hitting any site too often
• Be **Robust**: Be immune to spider traps and other malicious behaviour from web servers
  • Be careful to spams

What Any Crawler Should Do

• Be capable of **distributed** operation
  • designed to run on multiple distributed machines
• Be **scalable**: designed to increase the crawl rate by adding more machines
• **Performance/efficiency**: permit full use of available processing and network resources
• Fetch pages of “higher **quality**” first
• **Freshness/Continuous** operation: Continue fetching fresh copies of a previously fetched page
• **Extensible**: Adapt to new data formats, protocols
### Basic Crawler Architecture

- **WWW**
  - **DNS**
  - **Fetch**
  - **Parse**
  - **Doc FP's**
  - **URL filters**
  - **URL set**
  - **URL Frontier**

### Processing Steps in Crawling

1. Pick a URL from the frontier
2. Fetch the document at the URL
3. Parse the document
   - 1. Extract links from it to other docs (URLs)
4. Check if document has content already seen
   - 1. If not, add to indexes
5. For each extracted URL
   - 1. Ensure it passes certain URL filter tests
   - 2. Check if it is already in the frontier (duplicate URL elimination)
URL Frontier

- Can include multiple pages from the same host
- Must avoid trying to fetch them all at the same time
- Must try to keep all crawling threads busy

Explicit and Implicit Politeness

- **Explicit politeness**: specifications from webmasters on what portions of site can be crawled
  - `robots.txt`
- **Implicit politeness**: even with no specification, avoid hitting any site too often

```plaintext
User-agent: *
Disallow: /yoursite/temp/

User-agent: searchengine
Disallow:
```
- No robot should visit any URL starting with "/yoursite/temp/", except the robot called “searchengine”
URL Frontier: 2 Main Considerations

- **Politeness**: do not hit a web server too frequently
- **Priority/Freshness**: crawl some pages more often than others
  - Pages whose content changes often (e.g. News sites)
- These goals may conflict each other.
  - e.g., simple priority queue fails – many links out of a page go to its own site, creating a burst of accesses to that site.
- Even if we restrict only one thread to fetch from a host, can hit it repeatedly
- Common heuristic: insert time gap between successive requests to a host that is >> time taken in most recent fetch from that host

Summary

- History of Web search
- Basics of web search
- Usage of web search
- SEO
- Web crawling
Resources

- Text book 1: Intro to IR, Chapter 19
- Text Book 2: IR in Practice: Chapter 3
- YouTube Videos (nice to watch)
  - How Search Works. Google  
    https://www.youtube.com/watch?v=BNHR6IQJGZs
  - The Evolution of Search. Google  
    https://www.youtube.com/watch?v=mTBShTwCnD4
  - What Is The Deep Web?. Mashable  
    https://www.youtube.com/watch?v=UOK7aRmUtw